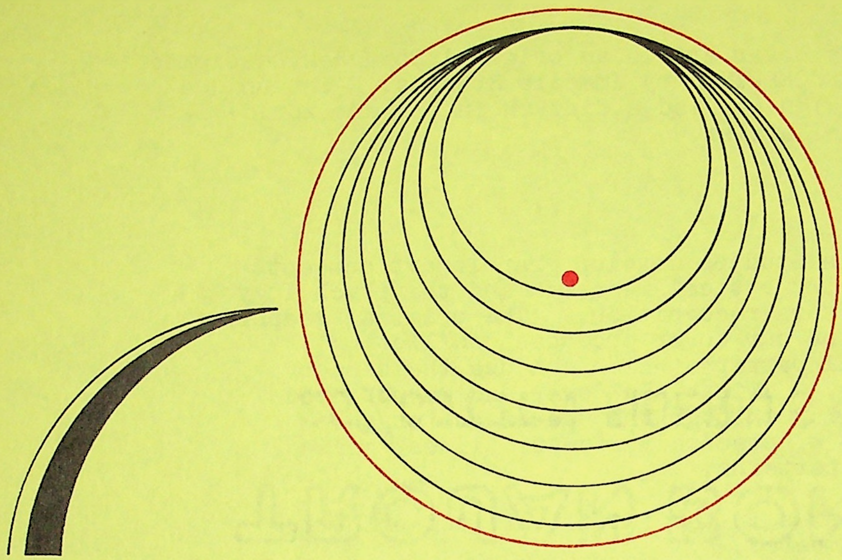
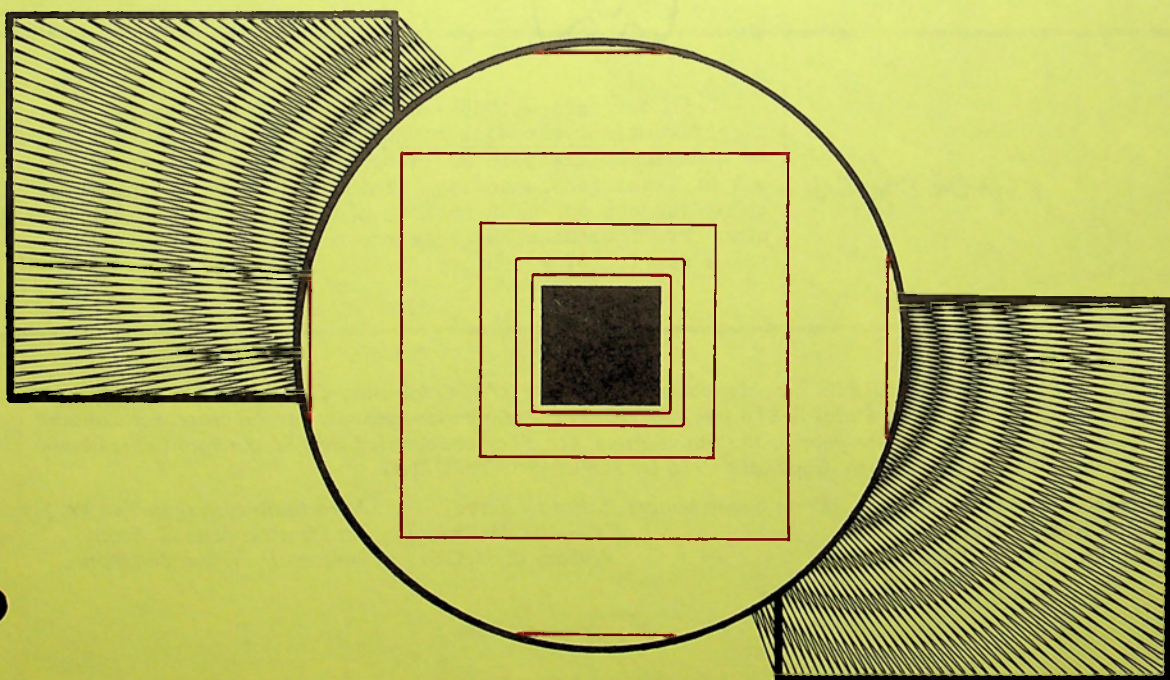
33
Popular Computing
VOL 3 NO 12 DECEMBER 1975





This month's cover art is an original Borávček, designed for POPULAR COMPUTING by Immedia Studios. The unique two-part design (called a diptych in the art world) is signed by the artist.

Immedia Studios develop graphic art concepts utilizing optical research and the psychology of pattern recognition. The original graphic print on our cover produces conflicting optical patterns which confuse the sensory organs. This visual overload is achieved primarily through the use of one or two layers of precisely repeated lines, called moire patterns.

All copies in the initial limited print run are numbered.

0800

Apologies

In our review (PC31-12) of David Thorndike's magnificent Encyclopedia of Banking and Financial Tables, we managed to misspell Mr. Thorndike's name--three times, but at least consistently. This is just about the worst thing you can do to an author, and particularly embarrassing since Mr. Thorndike has been known in computing circles for a quarter of a century. □

POPULAR COMPUTING is published monthly at Box 272, Calabasas, California 91302. Subscription rate in the United States is $18 per year, or $15 if remittance accompanies the order. For Canada and Mexico, add $4 per year to the above rates. For all other countries, add $6 per year to the above rates. Back issues $2 each.

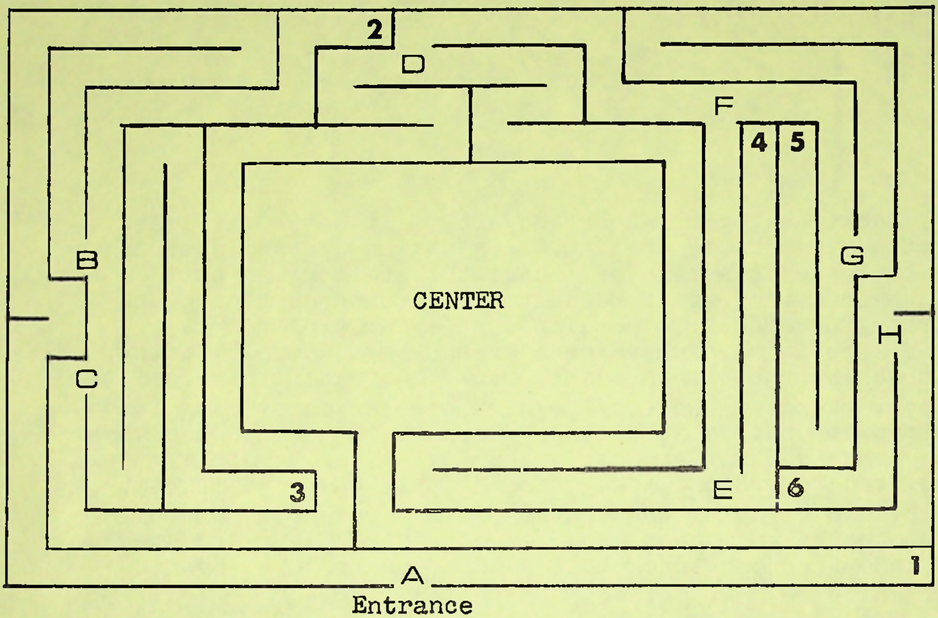
Backtracking

BY THOMAS R PARKIN

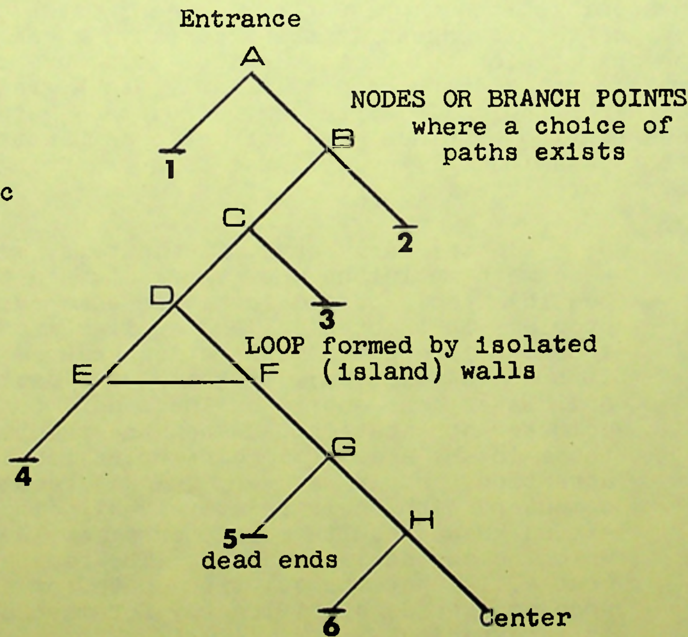
Backtracking is an arcane art. It dates, at least, from the time of the ancient Greek philosophers discussing ways to traverse a maze or labyrinth, stemming in part from their mythology of Daedalus' labyrinth confining the Minotaur for Minos. The simple rule for exploring a proper maze is to explore each branch to a stopping point, backtrack to the branch point, mark the branch traversed, and systematically select the next branch, continuing in this manner until a solution is reached. In case a branch being traversed has another branch in it, we have now discovered recursion, i.e., simply apply the same rule at that second (and any subsequent) level branch point until all the paths have been explored at a given level of branch point and then backtrack to the last higher level branch point which has not yet been exhausted.

In effect, we have converted the maze traverse problem into a mathematical graph which may not physically resemble the maze, but which is its logical equivalent. This graph has a tree-like structure with the root or apex representing the starting point and the ends of the branching paths representing dead-end paths or the path to the center, as shown in the accompanying diagrams. It is clearly easier to see the logical structure of the original maze in this graph, and, similarly, such graphs, drawn out or algorithmically expressed, allow the systematic description of other more intricate combinatorial problems to be examined. We shall use this branching-type logical structure later.

In the early days of computers, many of the programmers were mathematicians who eagerly turned to computers as possible tools to aid them with some quite intractable problems in theoretical and applied mathematics. The interest in applied mathematics spawned FORTRAN (from FORmula TRANslation), emphasis on floating point numbers, and, later, the entire business data processing area of computer application. Among the theoretical problems, those in the area of combinatorics received considerable attention, and it was very quickly recognized that even computers with their relatively blinding speeds could not extend known results more than a few levels beyond those which humans could produce. The reason for this, of course, is that the solution space for a combinatorial problem expands significantly for each new level to which it is extended.



Topological Equivalent of the Maze at Hampton Court (see also the diagram in Issue 32, page 10).



Graph Isomorphic to Paths of the Maze at Hampton Court

For example, the $26 \cdot 25 = 650$ two letter words which are possible with a 26 letter alphabet with no repetitions allowed expands to 7,893,600 five letter words and to approximately 1.9 times 10 to the 13th power ten letter words.

Although the principles were widely known and applied for some time by early programmers, the name backtracking was probably first used by Golomb to describe the programming technique when used to explore combinatorial problems of some complexity. If backtracking were only used to systematically explore all the possible solutions to a combinatorial problem, it would only be the mechanism for implementing the exhaustive, brute force enumeration of all cases at some level of the problem and in that respect, it would not have particular advantage over any other method. However, if one combines the principle of backtracking with two other requirements, it becomes a very powerful technique indeed.

These other two requirements are that a procedure or algorithm be formulated for the particular problem which will generate all possible end cases of interest in a branching tree-like order and that a test or tests be provided which can detect the dead-end nature of groups of end points at the highest possible branch level. The first of these requirements insures that the procedure being followed to explore the myriad possible cases will certainly generate all of them, and each of them only once, preferably. Furthermore, the procedure should be organized so that after each successive step or level of application of the algorithm for generation of cases, the remaining solution space of end points is further subdivided in an orderly way.

For example, the eight binary numbers which can be formed with three bits can be subdivided into two classes on the basis of the first bit: those beginning with zero (000, 001, 010, 011) and those beginning with one: (100, 101, 110, 111). This subdivision into classes may not always be quite so symmetrical; for example, the same eight numbers can be separated into two classes, those containing at least two adjacent zeros, and all others: (000, 001, 100), (010, 011, 101, 110, 111).

The second requirement is critical to the success of any practical real problem in combinatorics being put on a computer and the advantageous use of backtracking. What is needed is a criterion or test which can be applied at the highest possible level in the orderly generation of the entire solution space such that many of the exhaustively enumerable end cases can be eliminated each time the test is applied. If we can only test an end case for applicability to our desired goal or solution after it has been explicitly formed, then we must generate and test every one of the potential end cases and we have used brute force on the problem and possibly consumed great amounts of computer time.

On the other hand, if we have a problem where we are going to go, say, eight binary levels deep in generating our end cases, and we have a test which will allow us to reject all further effort along a branch after, say, three levels, we have, in general, saved ourselves 31/32nds of the total work of the program. In most practical cases of the application of backtracking, the applicability of our test is usually at a variable level; there may even be several criteria, hence tests, which can be applied, and the solution space is usually not generated simply with binary levels, but the number and multiplicity of the levels may be very large indeed.

Tests for detecting large classes of dead-end branches and the algorithm for generating the solution space are generally not independent, unfortunately. Therein, then, lies the essential trick of how to apply backtracking to a particular problem. Indeed, a further practical detail often intrudes; namely, how to code the objects of interest in the combinatorial problem so that they can be manipulated and tested easily in the computer. The principle of backtracking is quite simple: proceed until blocked; back up to an earlier branch point, and continue. The trick of applying backtracking usually lies in the orderly generation of cases to be tested coupled with the identification of appropriate criteria to detect the blockage and the sometimes fussy problem of coding representation of the objects of interest. Hence the first sentence of this essay: backtracking is an arcane art.

Next month we shall give a problem and show how backtracking is applied to its solution. □

Random Digit Generation by the Test-Passing Algorithm

Each new scheme for the generation of pseudo-random digits (or numbers) is validated by subjecting the output to eight standard statistical tests:

PC33-7

1. The frequency test. Counts are made of the appearance of individual digits; these form a 10-way distribution. The theoretical distribution calls for 10% of each digit. The observed values and the theoretical values are compared for goodness-of-fit by chi-squared, to show that the observed frequencies are close to, but not too close to, the theoretical.

2. The serial test. Counts are made of the appearance of the digits taken two at a time. This makes a 100-way distribution, for which the theoretical values should all be 1% of the total. Again, the comparison between the two sets of values is made using chi-squared.

3. The gap test. A distribution is made of the lengths of the gaps between successive appearances of the same digit. These gaps can be as small as one or can be very long. The mean value should be 10, and gaps of over 40 should be aggregated. The gaps are taken for all 10 digits. The observed distribution is compared with the theoretical by chi-squared as before.

- - tongue-in-cheek - -

4. The poker test. Taking digits four at a time, counts are made of the types: four of a kind; three of a kind; two pairs; one pair; and none alike. Compare the observed frequencies with the theoretical. The choice of four digits, rather than the five indicated by the name of the test, is solely due to tradition.

5. The maximum test. Taking successively generated digits three at a time, a count is made of those triplets for which the middle digit is greater than the other two. The triplets for which this is true should occur 28.5 percent of the time. As usual, one wants to be close to 28.5 percent, but not too close.

6. The D^2 test. This is a test of random numbers, rather than random digits. Random number generators usually produce numbers that are uniformly distributed between zero and one, considered as fractions. Two such random numbers can thus locate a point at random in the unit square, and the distance between two such points will range from zero to the square root of 2. The theoretical distribution of such distances is known (see reference 5).

7. The correlation test. As in test 6, random numbers are used to generate random points in the unit square. The square is divided into 100 equal smaller squares, and each of these squares should receive 1% of the points.

8. The coupon collector's test. This is again a digit test. For successively generated digits, counts are made of the number of digits that must be taken to obtain a complete set of all 10 digits.

(For example, in the sequence of leading digits of pi, it takes 33 digits before a complete set is obtained.) The length of any one string must be at least 10, but may be any length longer; strings of length over 40 are aggregated for statistical purposes. The theoretical frequencies for this distribution are known to some 35 digits of precision (see reference 6).

While each new algorithm attempts to optimize some computer trait (e.g., minimum execution time, minimum storage use, minimum number of instructions, etc.), it is clear that the logical attack is precisely backwards. The actual goal, however carefully concealed, is to pass those eight tests. It follows, therefore, that the ultimate method is one which capitalizes directly on the true goal; namely, an algorithm which is based on the tests themselves. Hence the derivation of the Test-Passing method.

The new algorithm is simply stated: at any stage, select for the next digit that one which will tend to make the total collection pass all eight tests. This is the theoretical definition. As is customary, we need also an operational definition: select that digit which will tend to correct that test which is most out of control at that stage.

Neither of the definitions provides a way to get started. Any existing generator can be used to produce, say, 400 digits as starting values; this is housekeeping for the method, and is done only once.

When a new digit is to be generated, the eight tests are applied to all the digits so far available. Suppose that the situation is as follows:

	Chi-squared	p
1. Frequency test	5.380	.80
2. Serial test	19.446	.76
3. Gap test	35.608	.24
4. Poker test	1.839	.72
5. Maximum test	5.412	.02
6. D^2 test	12.247	.19
7. Correlation test	31.410	.06
8. Coupon collector's test	22.685	.56

Clearly, at this point, the maximum test is out of bounds, so the next digit selected should not form a local maximum (and probably the next dozen points would be selected on that criterion). Eventually, the maximum test will be satisfied; that is, its probability will be raised to .05, at which time some other test will be the weakest, and so on. Tests 6 and 7 are the most awkward to manipulate, since they each require many digits to form one new test case. On the other hand, each attempt to bring them within bounds allows for the generation of many new digits, during which time the additional computation for the other tests may be suspended, thus saving compute time.

The Test-Passing algorithm, written as a subroutine for the 370/158, involves 1823 instructions and (on that machine) takes an average of 43,250 milliseconds to generate one new digit. Each of the eight tests requires some data storage, and all of them together require storing most of the chi-squared table. Total data storage comes to 1381 words. A new implementation, written specifically for efficiency, is expected to improve the above figures by at least 5%.

Experience in implementing the algorithm indicates that the poker test is the one that most frequently wanders off scale or, looking at it another way, continuous monitoring of the poker test best insures that all the tests remain stable simultaneously. Thus, in practice, the priority order for the tests should be as follows: 4, 3, 5, 8, 1, 2, 6, and 7. There is some evidence that if the tests are applied in that order, tests 2, 6, and 7 may never be used to dictate the choice of the next digit.

A delicate problem arises when two or more of the test criteria are at identically critical points. Even though they are being monitored in priority order, a choice must be made as to which test should be catered to for the next digit. The obvious solution is to make that choice at random, using some handy digit recently selected.

Note: A version of this article, with the lines of type carefully scrambled, appeared in Software Age, June, 1970. Flowcharts for the algorithm described here will not be furnished to anyone on request, and no source deck listing exists. Do not write for further information.

References

[1] Lehmer, D. H., "Mathematical Methods in Large-Scale Computing Units," *Annals Comp. Laboratory Harvard Univ.* 26 (1951) pp. 141–146.

[2] Forsythe, G., "Generation and Testing of Random Digits at the National Bureau of Standards," Los Angeles, Nat. Bur. Stand., *Appl. Math Series* 12 (1951) pp. 34–35.

[3] Arthur, Austin O., "Random Digit Generation," *Computing News*, V. 4, September 15, 1956, pp. 5–7.

[4] Kendall, M. G. and B. Babington-Smith, "Randomness and Random Sampling Numbers," *J. Roy. Stat. Soc.*, 101 (1938) pp. 147–166.

[5] Gruenberger, F. and A. M. Mark, "The d^2 Test of Random Digits," *Math. Tables Other Aids Comp.* 5 (1951) pp. 109–110.

[6] Greenwood, R., "Coupon Collector's Test for Random Digits," *Math. Tables Other Aids Comp.* 9 (1955) pp. 1–5, 224, 229.

[7] Hull, T. E. and A. R. Dobell, "Random Number Generators," *SIAM Review* V. 4, No. 3, July, 1962, pp. 230–254.

[8] Horton, H. Burke, "A Method for Obtaining Random Numbers," *Annals Math. Stat.* 19 (1948) pp. 81–85.

Random Digit Tables

In our series on the Art of Computing, essay number five (Issue 21, December 1974) discussed the generation of random numbers. An historical footnote concerns the construction of what was possibly the third table of random digits, circa 1944.

The first table was that of L. H. C. Tippett, in the 1920's, produced by taking numbers from census tracts in Great Britain. The second table was that of M. G. Kendall and B. Babington-Smith (1938), made by pulling digits one at a time from the position of a rotating wheel. The definitive table is the one of 1,000,000 digits produced at the RAND Corporation and published in 1955.

A need arose in the middle period, in connection with a problem in scrambling information punched in cards. A deck of cards could be scrambled effectively by means of the collating device available at that time as a special feature for the IBM 075 sorter. To use this device, the sorting brush was disengaged, and a rotary dial was set to a number between 2 and 13. The machine would then distribute the cards passing through it in rotation into from 2 to 13 stackers. To scramble a deck, the following procedure could be used. Set the dial to, say, 13. Start the deck through the sorter, and remove the cards from the stackers at intervals (e.g., whenever any stacker became nearly filled--this could be done without halting the machine) and reinsert them into the input hopper. From time to time, some of the stacked cards could be moved to a rack, and more of the original deck added to the stream. With diligence, and with all cards moving through the machine three or four times, the deck could be considered scrambled. The process is tedious and requires a modicum of intelligence by the operator to insure that no pattern of operation is superimposed on what should be a randomizing scheme.

The procedure worked, and a 100,000-card deck could be well scrambled, using two 075's, in an 8-hour shift, in the sense that any given card had an equal chance of being in any position in the final ordering.

If the cards to be scrambled could have punched on them some random digits, and these digits were properly produced (a notion that was not so well defined or understood in 1944), then the deck could be scrambled by normal sorting on the random digits. This procedure would be faster, and would lend itself to specific instructions to the operator.

This led to the need for a random digit table, and one was produced by careful use of the collating device, together with another device on the 513 reproducer; namely, a consecutive numbering option. This device could be reset to any specific 3-digit number, and would then number punch cards with consecutive 3-digit numbers.

A deck of, say, 3100 cards was numbered from 001 through 100 in columns 1 to 3. The deck was then thoroughly scrambled, using the collating device on the sorter, and was punched with consecutive numbers in columns 4 to 6, starting with 101. This process was repeated 26 times, until 78 columns were filled with digits. The resulting table of nearly a quarter million digits would not pass most of today's standard tests of randomness, but was quite adequate for its intended task--that of scrambling a large deck.

Some years later, a suggestion of H. Burke Horton provided a means of refining the crude deck described above into a larger deck of better quality. Horton showed that the addition of decimal digits without carry yielded new digits that would test better for randomness than the originals. Thus, a tabulator-summary-punch combination could be used to enlarge a random digit deck and improve it at the same time. The input digits were fed to counters in the tabulator, and summary punched out of the same counter position, thus suppressing the carry. A 2-wheel counter could handle a single digit. In a 4-wheel counter, the input digits could be fed to the low and high order wheels; a carry would propagate across the counter, on the average, every 222 cards, but this small perturbation could be ignored. In one pass, 26 digits could be summed, and 26 columns of a new deck be punched. If a summary card was punched for every 10 cards moving through the tabulator, each run would expand the deck by 1/3 of 1/10 its size. Thus, given an original deck of 3100 cards, 310 new cards could be produced, punched also with 78 columns, in about an hour.

It is possible that a 5000-card deck produced by this scheme at the University of Wisconsin Computing Center in 1950 was the fourth random digit table in the world.

The table of random digits on the next two pages was calculated by more modern techniques by Lee Armer. The generator shown in PC21-8 was used, and several calls were made of the subroutine for <u>each</u> <u>digit</u> of the table.

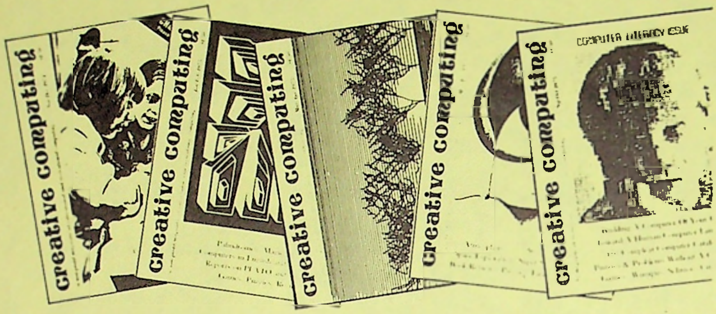
PC33-12

79063	85834	99900	15631	62956	16288	31161	58936	08393	34210
28124	54783	02423	88659	34237	24720	52228	48020	70566	02087
05822	10073	07413	09603	21622	90134	85267	08279	31605	02373
40085	24117	10266	84944	66051	57637	52075	53485	76627	25933
90784	47631	44436	00377	03832	78039	24529	81072	05606	96220
51571	32647	58951	31962	35167	48606	66082	10618	82781	17628
79099	37485	65035	25319	46805	35454	60268	68867	32417	30469
64843	55605	15861	98937	64300	66214	65386	08070	19490	09340
74451	71882	46174	44667	91448	02663	70435	94615	60783	83586
01825	67227	91935	31065	27854	85867	25357	91599	99314	09399
92560	09288	11776	84245	87435	00795	09782	85766	68250	24390
51905	53342	73954	26455	36498	12098	61834	87535	52543	83205
92868	35565	17349	00004	16945	51390	08933	32947	25654	85722
38633	95067	27128	62078	72532	34798	52207	73027	68800	90872
21159	46773	54773	31753	26797	86967	53117	58756	85592	86847
27400	16046	82360	95276	41701	68605	71089	99649	41644	53482
70288	01149	57468	60535	75347	16218	02363	08334	36093	80717
98690	25347	13640	24727	91660	62081	97977	78221	93247	12831
83528	72269	58067	78925	03334	52395	91430	39610	06457	90839
97684	05035	40450	53730	80848	75148	94319	90862	04684	52922
68218	73591	34563	87159	68353	35897	05829	55750	80074	83325
90568	72164	36598	39166	07867	00677	28736	81898	95692	92019
46798	91977	75775	44162	14698	91625	49119	25621	65228	58496
46604	23851	30533	22816	64761	29098	17401	44387	70232	08189
13060	65554	46631	91761	74188	51384	99175	41323	27233	85537
30048	03680	61454	43059	94552	76041	29121	80006	49646	79594
65534	23399	27590	47237	70419	39987	87256	72924	15106	69117
41500	99029	48500	22206	75822	29825	24523	01673	19544	50307
92128	30488	04554	61400	35836	78906	38458	66228	98786	37221
61023	35073	14559	65898	55253	49601	20049	34927	01624	43465
91910	29073	96070	85544	72672	38834	07969	60686	19658	86398
06250	22497	14896	82704	60515	71721	09191	84532	91076	93649
89634	44464	21948	76618	10185	98043	68895	65493	17247	49544
81587	53412	72725	17352	99123	24667	57718	53775	37042	78131
36975	17733	82803	84390	28471	41013	27120	69121	54878	36412
51215	64890	70512	08510	62812	82532	59598	18578	87292	07727
56423	02151	83969	09394	39264	13974	11832	50817	53415	36678
65056	84844	66293	88552	31442	03791	54818	79021	25194	02920
79039	54854	75663	98779	45559	44595	02243	65928	90335	68402
00503	79003	25697	84687	85234	11660	64301	10776	93259	61409
54960	43163	53298	77361	66500	24515	53746	82821	32678	55975
22780	93859	97645	14993	97277	50799	26843	05090	24410	16391
02191	56194	63024	69646	73891	07240	46015	38167	71765	06687
62513	99347	39906	49238	73099	29475	32048	79755	03485	67550
71118	32272	98705	49195	97531	17804	74607	93406	27180	87198
46174	39646	96669	40070	60986	41166	39886	92955	17877	57500
64849	53101	74459	25736	89128	82661	62478	34872	10531	03854
66164	42205	74950	45558	35977	34051	27290	74587	30138	96196
16376	60057	58541	47096	67470	26928	75190	17483	15150	67375
58358	72711	19213	87617	36352	05322	91333	80721	82152	88330

A Table of Random Digits

57578	28888	92720	17771	30768	54254	59606	27926	58397	02953
82332	81549	99161	82296	97530	64139	20267	14908	84762	23207
81938	01963	34148	33334	85143	68496	46611	07341	34480	43466
34264	94724	54222	85545	39420	29707	19673	32154	32030	50570
28528	58873	18439	61969	45805	70511	95426	12483	59766	18791
23794	46204	44113	52714	61320	90858	11432	49167	66737	34286
22970	09102	68718	42066	04804	85245	61875	87946	09169	73542
22576	08825	69839	94091	36033	85459	52246	26804	20620	46584
23186	32410	95069	08205	62235	46972	83220	02049	41945	27005
96460	07477	83050	90223	65967	78894	07573	26174	06266	92041
98399	55376	22532	54113	02434	98488	05473	85776	36099	32699
76768	60693	68336	08283	79885	61028	10012	69278	64253	46802
72242	97717	65468	42839	00649	90579	76533	55965	64293	69936
67605	43636	62867	31540	56380	52838	40989	59133	42593	61809
45788	27744	68014	83507	76153	01071	51378	78867	38128	39342
82789	47182	77298	04324	58028	65960	43692	02966	03006	23814
78685	16588	63209	60243	26411	67926	72620	31643	54940	59788
78951	99032	07066	36323	31745	07875	03223	03948	42386	60829
23834	29773	35252	29438	54961	71187	75538	90155	82215	09388
17711	41696	30256	77862	29936	06725	46573	03698	48665	07249
14603	11847	73382	57184	57962	70905	93849	62659	00705	04632
61123	65759	15745	42971	41067	10338	76769	79949	44740	00637
85758	65367	84000	01056	45764	37541	15477	80548	77726	41057
19524	41685	80553	41976	70131	26985	28888	07530	57199	02693
47084	55366	92922	60873	04586	43831	64685	04761	66024	57784
40065	25479	30931	92827	97720	43273	32054	31911	78112	39838
98316	19308	70742	80748	71515	92966	16996	44054	65132	87184
59608	10577	18052	44274	70388	86025	49016	27759	04717	34969
27384	77258	38401	57985	96824	56892	24208	13950	30551	70246
67207	72240	82819	33517	22777	56591	77511	47202	29768	76680
73202	48115	10215	90106	20871	35180	02599	73880	56665	45099
27223	67927	02843	36522	49335	50357	98731	35574	78313	53023
46255	26416	18393	22047	91609	54564	03374	06711	86580	50287
69628	56818	89514	49159	11195	07760	18586	13545	42587	71692
23987	60498	18846	82629	07032	69718	91729	41879	88614	42089
66459	53832	13738	51135	08400	99706	57668	10462	61346	72946
29663	41719	77702	67872	93250	86461	33262	57835	77490	97306
93120	84823	58590	06856	56523	33319	41042	33951	12753	25512
39751	78153	25119	16023	02232	14971	96510	55702	88222	79929
63904	52751	15850	42302	10799	68565	61921	12426	14871	59912
51192	84182	75614	92487	15873	54438	39679	20696	52161	32211
61792	24812	98165	25668	28921	48634	50551	55132	89157	94279
32551	54605	21556	34462	84652	09495	61897	79736	14147	27507
42734	24421	26377	88226	70692	95687	85437	73312	72359	26146
56074	87786	97777	99318	86209	55512	23581	00512	96694	45732
60590	91267	69410	98390	37183	07762	84385	60722	21784	93900
76770	26162	78755	37371	39789	41732	12279	30887	18125	23862
47468	82363	03989	28008	63864	57399	88915	14940	62954	06664
43144	93105	13302	10242	01488	94564	66753	54258	46481	46977
61646	89892	47997	14203	86719	73439	40692	54781	29929	78686

Can You Cope With The Computer Age?



WHAT'S THE BEST BASIC BOOK IN PRINT?

Did you know there are 36 BASIC language textbooks today? Which is best for 7th graders? For college freshmen? For an experienced FORTRAN programmer? What kind of games or books are best to teach computer literacy to 4th graders? What book has the best discussion of the computer threat to society? Read *Creative Computing* for the reviews that pull no punches.

* * *

THE NEXT PICASSO — A COMPUTER?

Can a computer create original art? Or is it just a tool? Does all computer art look "mechanical"? Will computer art have an impact on art as a whole in the future? Can you imagine a printed circuit board winning a blue ribbon in an important art exhibition? Find out more in *Creative Computing,* the magazine that brings computer art to *you.*

All subscribers will receive a $5.95 computer art book FREE!

NO COMPUTER TODAY?

Looking for learning activities that don't need a computer? Things that teach about computer concepts, the role of the computer in society, its use as a tool in math, science or social studies? How about a whimsical centerfold poster for your bulletin board every issue? Comics too? It's all in *Creative Computing,* the down-to-earth computer magazine.

* * *

WHAT'S THE LARGEST INDUSTRY IN THE U. S.?

Automobiles? Steel? Agriculture? Not any more. It's computers. More dollars. More jobs. More impact. What do you know about computer careers? Where do you start? What courses to take? Where are the opportunities? Get the answers in *Creative Computing* so you can get the jobs.

DO YOU ENJOY PUZZLES?

Do you like mathematical diversions? Are mind benders your bag? A printer uses 1215 characters to number the pages of a book — how many pages in the book? What's the next number in the sequence 63, 94, 46, 18,? What digit is represented by each letter: HOCUS + POCUS = PRESTO. You'll find lots more in *Creative Computing,* the fantastic puzzles and pastimes magazine.

* * *

COULD A COMPUTER TAKE OVER THE WORLD?

Isaac Asimov in a new short story describes what happens when all the computers on earth after a nuclear holocaust link up to support the few remaining human survivors. Want to know the outcome? Then get *Creative Computing,* the magazine that speaks *your* language.

Try *Creative Computing* for a year — only $8.00. Or three years for $21.00. Or send for a sample issue for $1.00. Please include money; otherwise we add a $1.00 billing charge.

☐ 1 Yr. $8 ☐ 3 Yrs. $21 ☐ Sample $1

NAME ____________________

ADDRESS ____________________

CITY ____________________

STATE __________ ZIP __________

Return to *Creative Computing* P. O. Box 789-M, Morristown, N. J. 07960.

Log 33	1.518513939877887478045227874498139550906831054657149
Ln 33	3.496507561466480235457188814887655004469197411760167
$\sqrt{33}$	5.744562646538028659850611468218929318220264457982792
$\sqrt[3]{33}$	3.207534329995826487552515171719520111361851663360572
$\sqrt[10]{33}$	1.418572034507080759397456853588452717019865422289856
$\sqrt[100]{33}$	1.035583541049424315467280579465702930458525446993546
e^{33}	214643579785916.06462429776153126088036922590605479789725918541262650031306985868251115524
π^{33}	25465124213045828.4705835412063330235579541312710788154733947167377636486801395690786595
$\tan^{-1} 33$	1.540502566876121517825521698713981316580997003141962

N-SERIES 33

Jefferson's Cipher Device

(Factual material on cryptography in this article is taken from the unique and superb book The Code Breakers: The Story of Secret Writing, by David Kahn, The Macmillan Company, 1967, 1164 pages.)

One of the most significant advances in the science of cryptography was made by Thomas Jefferson around 1790 while he was Secretary of State. Jefferson invented a cipher device, consisting of a number of wheels free to turn independently on a common axle. The flat rims of these wheels are evenly divided into 26 parts and engraved with the alphabet, randomly permuted differently on each wheel. A set of 50 such wheels are made available to the sender and receiver of cipher messages, and 25 of them are used at any one time. The choice of which 25 can form the secret key for the system, to be decided in advance of any period of use. The choice of the order in which to use the 25 wheels forms the key for an individual message.

With the 25 wheels mounted in the proper order on the axle, the first 25 letters of the plain text message are aligned on the wheels. Then any other 25 letters, found by rotating the wheels together, constitute the cipher text to be transmitted.

Decipherment requires the same wheels in the same order. The 25 cipher text letters are aligned on the wheels, and the assembly is rotated to find the set of 25 letters that make sense.

It will be noticed that Jefferson's wheel cipher (and its modern derivatives) differs from all other cipher systems in two respects:

1. It requires intelligence to use it. For example, it would be difficult for a person who knows no German to decipher a message in German, even though he has both keys. Thus, one of the requirements of military cryptography, that a system be operable by low-grade personnel, is not fulfilled.

2. The system cannot be used to transmit meaningless information. In particular, it is impossible to do double encipherment with the system (i.e., encipher a message with one set of keys and then encipher the result with another set of keys).

As a consequence, the system must be hand operated, and does not lend itself to automatic procedures. It would be difficult (but not impossible) to program the system for a computer. Kahn points out:

"Later, other branches of the American government used the Jefferson system, generally slightly modified, and it often defeated the best efforts of the 20th-century cryptanalysts who tried to break it down! To this day [1967] the Navy uses it. This is a remarkable longevity. So important is his system that it confers upon Jefferson the title of Father of American Cryptography. And so original is it that it sets Jefferson upon a pedestal far more prominent than those accorded to men like Vigenere and Cardano, whose names are usually thought to be household words in the history of secret writing."

Consider now the pattern labelled S. The numbers shown are on 14 wheels, as in the Jefferson device. The wheels were originally set to contain ten binary numbers, read from left to right across the wheels. The ten numbers are familiar constants with the same position for the binary point for each number. For example, if one of the ten numbers was pi, the sequence would be some 14 positions of the sequence:

00011001001000011111

(i.e., the binary version of pi).

After aligning the ten numbers, the wheels were rotated independently to the positions shown in S.

Problem: rotate the 14 wheels so that the original ten numbers can be identified.

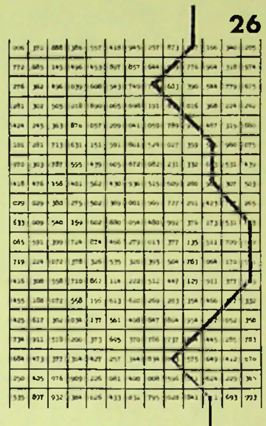
PROBLEM 111

S

1	1	1	1	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	1	0	1	0	1	0	0	1	1
0	0	1	0	1	0	1	0	1	1	0	1	0	1
0	0	0	1	0	1	1	0	1	0	0	1	0	0
0	0	1	0	1	1	0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0	1	0	1	0	1	0
0	1	0	0	0	0	0	0	1	0	0	0	1	0
0	1	1	0	1	1	0	1	0	1	0	1	0	0
0	0	1	0	0	1	1	1	1	1	1	0	0	1

Contest 1 — —The Outcome

26



Red Line — Blue Line

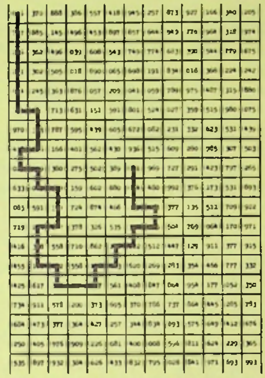
The first POPULAR COMPUTING contest appeared in issue 26 (May 1975). Given an array, 13 by 19 cells, containing random 3-digit numbers, the task was to find a path from one side to the other having the greatest sum of the contents of the cells passed through. The solution to this problem is shown here. 44 solutions were received, of which 40 were correct.

Many contestants wrote lengthy analyses of the problem. The following is from Thomas R. Parkin, La Jolla, California:

> Someone has no doubt pointed out by now that in spite of the 400 billion (+) paths, the problem reduces to exactly 247 additions with a choice from among 2 or 3 addends for each augend; i.e., 703 cases. Interestingly enough, the algorithm is both deterministic and provable (by induction).
>
> Algorithm (as problem is defined): Start at the second row from the bottom and for each column select the largest of the two or three numbers in the bottom row which can be added in that column and add it to the number in the second row. (Note: we now define this second row with new numbers in it as the bottom row). Repeat until the original top row is used. Pick the largest total by inspection of the 13 totals. To determine the path by which this largest total is obtained, a record must be kept in a 13 x 18 array of where each succeeding new row of totals is obtained.
>
> It is interesting to note that the same path and the same maximum total will be obtained by proceeding from the top down, but that, in general, none of the other totals will be alike for the two directions. Incidentally, performing this reverse direction exercise furnishes another proof of the algorithm.
>
> The total of the 247 numbers in the array is 121575 for an average of 492.2. Thus, an expected total might be 9351.8, while the actual total of 15573 appears to come from an average of 819.6 for each of the 19 summands. Remarkable, considering the dispersion of from 1 to 993.

Since there was only one prize to be awarded, a tie breaking contest was sent to the 40 people who had submitted correct entries. Using the same grid of numbers as in the original contest, the new problem was to proceed from the cell in row 9 column 7 (which contains 001) to any of the corner cells, moving only horizontally or vertically without reentering any cell and without having the path cross itself. The object was to find a path for which the average contents would be the smallest; that is, the sum of the cell contents divided by the number of cells was to be minimal.

26



Red Line – Blue Line

The tie-breaker was won by Adelin Mekeirle, Brussels, Belgium, and his solution (33 cells totalling 7888 for an average of 239.030303) is shown. The winner receives his choice of $25 or a two year subscription to POPULAR COMPUTING.

The tie breaking contest problem also elicited comments from Thomas Parkin:

> The problem is quite different from the original problem and, as far as I know, somewhat intractable for computer solution. The only deterministic algorithm I know of is exhaustive enumeration of all possible paths. This is a combinatorial problem of high order yielding perhaps of the general order of 10^{100} paths to examine. (Whereas the total number of shortest paths along a rectangular grid from one point to another is easily calculated, I know of no simple way even to calculate the total number of paths of any length from some interior point to any of the four corners of a finite grid, let alone taking into consideration the weights of the path elements.)
>
> It certainly would be possible to devise an heuristic program which takes a given path and tries to improve it over some locally limited domain, or which exhaustively explores, say, a k x k region about the end of a tentative path, then choosing the most likely extension by one square and repeating the limited search. Unfortunately, people as yet don't know how to program computers to be able to employ the gestalt kind of approach whcih the human brain uses on this kind of problem. I have no doubt that, when computers are fast enough and we are clever enough to program them, we will achieve the practical approximation of the human eye/brain combination in dealing with two-dimensional problems. Perhaps I am saying just that I don't know how to tell a computer: "choose a few likely looking paths and then look around and see if you can improve them."